AD-785 676

# RESOURCE ALLOCATION AND SCHEDULING IN BALLISTIC MISSILE DEFENSE ADAPTIVE CONTROL SYSTEMS

Doyce E. Satterfield

Army Advanced Ballistic Missile Defense Agency
Huntsville, Alabama

1974

# RESOURCE ALLOCATION AND SCHEDULING IN BALLISTIC MISSILE DEFENSE ADAPTIVE CONTROL SYSTEMS

DOYCE E. SATTERFIELD
THE ADVANCED BALLISTIC MISSILE DEFENSE AGENCY
HUNTSVILLE, ALABAMA



## INTRODUCTION

The radar and computer system can be considered as the eyes and brain of a ballistic missile defense (BMD) system. Given no limitation on resources, a ballistic missile system's operational philosophy may be to utilize predetermined battle plans for the allocation of radar and computer resources. However, it is rarely economically feasible to provide excessive resources, and, since the defensive and offensive situation changes dynamically, it becomes imperative that the allocation of such resources adapt to these changes.

This paper treats the radar and computer system as an adaptive control system in which the allocation of the computer preprocessing rate, radar pulse rate, and computer postprocessing rate are varied dynamically. To determine the optimal real-time allocation of the radar and computer resources, the problem was first formulated mathematically and then solved in two phases.

## PROBLEM FORMULATION

To develop quantitative as well as qualitative results, a typical BMD system problem is considered. The principal physical elements of the problem are a radar, a computer, interceptors, and threatening reentry vehicles with decoys. The radar and the interceptor battery are under the control of a computer. The problem is then to allocate the two major resources, i.e., the radar and the computer, to the task of searching for potential future threats, tracking identified objects, launching interceptors, guiding interceptors,

tracking interceptors, and processing all related data. A typical objective of this allocation is to maximize the probability that the radar survives the attack. Any similar criteria, subject to the constraints on the available resources, such as minimizing the probability of a reentry vehicle penetrating the defense, may also be considered as the objective.

The measurement of the achievement of the objective of the problem can be considered a function of the following variables:

- The number of radar pulses allocated to searching
- The number of objects presently detected and the possibility of detecting future objects
- The number of radar tracking pulses allocated to the ith object
- The impact points of the threatening objects
- The possibility of an object being a decoy
- The reliability of the interceptor
- The possible intercept points of the engagement
- The number of interceptors allocated to each target*
- The number of radar guidance pulses allocated to the jth interceptor which has been committed to the ith object
- Natural and man-made environmental effects.

Based on the sample problem, i.e., to maximize the probability that the radar survives the attack, the control variables of the problem are as follows:

- The search rate for future objects, F
- The future track rates of the n objects presently detected, $K_1, \ldots, K_n$
- The number of track and guidance pulses for each interceptor assigned to the n objects, $L_1, \ldots, L_n$

The major constraints of the sample problem are the available radar power, the computer processing speed, and the scheduling of $K_1, \ldots, K_n$; $L_1, \ldots, L_n$.

The sample problem can then be stated as follows:

Maximize: (0) Probability the radar survives = $f(F, K_i, L_i)$

Subject to: (1) $a_{11}F + a_{12}\left(\sum_{i=1}^{n} K_i\right) + a_{13}\left(\sum_{i=1}^{n} L_i\right) \le b_1(t)$

---

*Each target is assigned a single interceptor.

$$(2)\quad a_{21}F + a_{22}\left(\sum_{i=1}^{n} K_i\right) + a_{23}\left(\sum_{i=1}^{n} L_i\right) \leq b_2(t)$$

(3) Scheduling of $K_1, \ldots, K_n; L_1, \ldots, L_n$ subject to time frame limitations,

where $b_1(t)$ = Maximum amount of radar energy available per second
$b_2(t)$ = Maximum number of basic computer instructions available per second
$a_{11}$ = Radar energy required for a single search pulse
$a_{12}$ = Radar energy required for a single target radar track pulse
$a_{13}$ = Radar energy required for a single interceptor radar pulse
$a_{21}$ = Number of basic computer instructions required to process each search pulse
$a_{22}$ = Number of basic computer instructions required to process each target track pulse
$a_{23}$ = Number of basic computer instructions required to process each interceptor track pulse

The probability of radar survival (Equation 0) must be expressed quantitatively as a function of the search, discrimination, and tracking algorithms, and other associated functions. Thus, given a particular BMD system's characteristics, one can decompose the probability of survival into various subfunctions which can be determined analytically.

The equations of the first two constraints are formed straightforwardly; however, the third constraint, i.e., scheduling, has a nonlinear, time-varying nature and cannot easily be expressed in an analytical form. Therefore, it would be difficult to solve the problem directly by considering all three constraints simultaneously, but it is possible to separate the problem into two phases. In Phase I, the allocation problem is solved without considering the third constraint, i.e., assuming that the transmitted and receive radar and computer times are contiguous. In Phase II, the required computer and radar times that were determined in Phase I are scheduled such that the overall schedule time is minimized.

## PHASE I

Phase I is used to establish the optimal blend of resources for the problem formulation which, as stated previously, is:

Maximize: (0) Probability the radar survives = f (F, $K_i$, $L_i$)

Subject to: (1) $a_{11}F + a_{12}\left(\sum_{i=1}^{n} K_i\right) + a_{13}\left(\sum_{i=1}^{n} L_i\right) \leq b_1(t)$

(2) $a_{21}F + a_{22}\left(\sum_{i=1}^{n} K_i\right) + a_{23}\left(\sum_{i=1}^{n} L_i\right) \leq b_2(t)$

$F,\ K_i,\ L_i \geq 0$

and is solved using the nonlinear, constrained search method denoted as the Complex method [1].

The following example problem with n threatening objects was solved by the Complex method programmed in FORTRAN and executed on a Control Data Corporation 7600 computer system. Although the problem is somewhat smaller than would be encountered in practice, it does exhibit the typical form of real-life problems.

The performance criterion for measuring the achievement of radar survivability can be expressed as:

$$Z = P_D(F)\left\{\prod_{i=1}^{n} P_T(K_i)\, P_I(L_i)\right\}$$

where

$P_D(F)$ = Probability of detecting all future threats, as a function of search rate F

$P_T(K_i)$ = Probability of survival against the ith object attacking the radar, as a function of object track rate $K_i$

$P_I(L_i)$ = Probability of survival against the ith object attacking the radar, as a function of track rate $L_i$ of the interceptor assigned to ith object

n = Number of threatening objects

The assumed typical performance functions are:

$$P_D(F) = 1.0 - 0.8 \exp(-F/1.5)$$

$$P_T(K_i) = 0.95 - 0.7 \exp(-K_i/2.5)$$

$$P_I(L_i) = 1.0 - 0.6 \exp(-L_i/3.5)$$

These functions are both monotonic (more resource produces more performance) and concave (more resource means that incremental performance gain decreases) and therefore, it can be shown that the necessary and sufficiency conditions for a unique global optimum are satisfied. The problem, with assumed values for radar energy per pulse, computer instructions per pulse, and total radar and computer resources, becomes:

Maximize: $$Z = \{1.0-0.8 \exp(-F/1.5)\} \{0.95-0.7 \exp(-K/2.5)\}^n \{1.0-0.6 \exp(-L/3.5)\}^n$$

Subject to: $$0.1F + n(0.02K) + n(0.003L) \leq 1.0$$

$$1.0F + n(0.2K) + n(0.3L) \leq 20.0$$

$$F, K, L \geq 0$$

$$n = 1, 2, \ldots, 10$$

The solution obtained is given in Table 1, along with the computer execution time for each solution.

## PHASE II

Phase II is used to establish the optimal schedule of the allocation results obtained in Phase I for the radar pulse rate and computer processing rate so as to minimize total system processing time. The problem of optimally scheduling tasks (n pulse groups) through the computer and radar is treated as a "two" machine problem with feedback That is, tasks are first passed through the computer (machine 1) for preprocessing, to the radar (machine 2) for transmission and reception, and then back to the computer for postprocessing.

This scheduling problem is a special case of the flow-shop sequencing problem which has been the topic of considerable research in the past few years [2][3][4][5] 6][7]. However, because of the combinatorial nature of the problem (n! combinations) which results in large storage requirements and/or long computational times, most work

has been limited to small problems or to heuristic methodology. To illustrate the immense size and nature of the scheduling problem, consider a 20-job problem. If one could evaluate the solution to each permutation (there are 20!) in a picosecond, it would take more than 28 days to try all possibilities.

Another solution method using a mixed integer formulation modified for evaluation with the branch-and-bound procedure has been investigated [7]. This method produces a linear programming problem, with from 3n to a maximum possible 3(2n-1) constraints and with 3n+1 variables, that must be evaluated in order to obtain a solution at each branch-and-bound node. It is clear that this method is not suited for large problems because of both storage requirements and computational time requirements. Other methods and examples of the scheduling problem can be found in Reference [8].

The two-machine problem with feedback is a special case of the flow-shop problem, but it retains enough similarity that heuristic scheduling appears to be the most feasible method of approach. Two heuristic algorithms have been developed especially for solving this scheduling problem and they hold for the following assumptions:

- All jobs and machines are simultaneously available.
- All jobs are of equal importance.
- No jobs can be processed simultaneously by more than one machine.
- No machine can simultaneously process more than one job.
- Process times are independent and deterministic.
- Transportation time is either included in the process time or is negligible.

Now, define the process time of job $\underline{a}$ on machine $\underline{m}$ to be $t_{a,m}$ for the first round of job $\underline{a}$ and $t'_{a,m}$ for the second round. Also, consider a partial schedule $\underline{\sigma}$ (which may be null) and a job $\underline{a}$ which, when augmented to $\underline{\sigma}$, yields the schedule (perhaps incomplete) $\sigma a$. Now, let $T(\sigma a,m)$ be the completion time of partial schedule $\sigma a$ at machine $\underline{m}$ (first round) and let $T'(\sigma a,m)$ be the completion time of the partial schedule (second round) on machine $\underline{m}$. (The completion time of the last job on the last machine is often called the MAKESPAN.)

The first heuristic algorithm (Method 1) developed for solving this scheduling problem will now be described.

Step 1. For each job i, calculate $f(i) = t_{i,1}/t_{i,2}$

Step 2. Arrange the jobs in alternating order of f(i), beginning with the smallest f(i). Let the complete schedule n so generated be

$S = (a_1, a_2, \ldots, a_n)$.

Step 3. Calculate the completion times $T(a_i,1)$ and $T'(a_i,2)$.

Step 4. If $T(a_n,1) + t'_{a_1,1} \geq T(a_n,2)$, indicating that the schedule obtained is optimal, go to Step 11; otherwise, continue with Step 5.

Step 5. Calculate $A = \sum_{i=1}^{n} (t_{a_i,1} + t'_{a_i,1})$

and $B = \sum_{i=1}^{n} \left(t_{a_i,2}\right). + \min \left(t_{a_i,1} + t'_{a_j,1}\right)$

Step 6. If $A > B$, go to Step 11; otherwise, continue with Step 7.

Step 7. Find the smallest $t_{a_i,1}$ and the smallest $t'_{a_j,1}$.

If $i \neq j$, set job $a_i$ first in the schedule and $a_j$ last, and then go to Step 10; otherwise, continue with Step 8.

Step 8. Find the smallest $t_{a_k,1}$ $(k \neq i)$ and the smallest $t'_{a_f,1}$ $(f \neq j)$.

Step 9. If $t_{a_i,1} + t'_{a_f,1} \leq t_{a_k,1} + t'_{a_j,1}$, set job $a_i$ first in the schedule and $a_f$ last; otherwise, set job $a_k$ first in the schedule and $a_j$ last, and continue with Step 10.

Step 10. Arrange the remaining jobs in alternating order of f(i), beginning the second scheduled job with the smallest f(i). Let the final schedule n so generated be

$S = (a_1, a_2, \ldots, a_n)$.

Step 11. Calculate the MAKESPAN of the schedule so obtained.

The second heuristic algorithm (Method 2) developed for solving this scheduling problem is now described*.

Step 1. For each job i, calculate

$$f(i) = \frac{\text{sign } (t_{i,1} - t_{i,2})}{\min (t_{i,1}, t_{i,2})} \ .$$

Step 2. Arrange the jobs in ascending order of f(i). Let the complete schedule n so generated be

$$S = (a_1, a_2, \ldots, a_n).$$

Step 3. Calculate the completion times $T(a_i,1)$ and $T(a_i,2)$.

Step 4. If $T(a_n,1) + t'_{a_1,1} \geq T(a_n,2)$, indicating that the schedule is optimal, go to Step 6; otherwise, continue with Step 5.

Step 5. Find i such that $T(a_i,2) \geq T(a_n,1)$. Arrange jobs i + 1 through n in ascending order of $t_{i,2}$. Let the final schedule n, as obtained, be

$$S = (a_1, a_2, \ldots, a_n).$$

Continue with Step 6.

Step 6. Calculate the MAKESPAN of the schedule so obtained. Repeat Steps 4 and 5 until no further improvement is possible or until an optimal solution is obtained.

*Development of this algorithm was based on suggestions offered by Dr. J. N. D. Gupta.

## ALGORITHM COMPARISON

A branch-and-bound algorithm has been programmed for comparing the effectiveness of the two heuristic scheduling algorithms. As stated earlier, the branch-and-bound method requires large storage and long computational time. However, it was felt that for the class of problems that it could handle some inference could be made as to the efficiency of the heuristic methods.

The branch-and-bound method developed to solve this scheduling problem is an adaptation of one by Lomnicki [2]. Each node in the branch-and-bound method represents a sequence of from 1 to n jobs. Consider node P, corresponding to the partial schedule $\sigma_r$ where $\sigma_r$ contains a particular subset (of size r) of the n jobs. Then a lower bound on the MAKESPAN of all schedules that begin with sequence $\sigma_r$ is:

$$LB(\sigma_r) = \text{MAX} \begin{bmatrix} T'(\sigma_r,1) + \Sigma_{\bar{\sigma}_r}\left(t'_{a_i,1}\right) \\ T(\sigma_r,2) + \Sigma_{\bar{\sigma}_r}\left(t_{a_i,2}\right) + \min_{\bar{\sigma}_r}\left(t'_{a_i,2}\right) \end{bmatrix}$$

where $T'(\sigma_r,1)$ and $T(\sigma_r,2)$ are the completion times for the last job of the r jobs in the sequence on machine 1 (second round) and machine 2, respectively, and $\bar{\sigma}_r$ is the set of n-r jobs that have not been assigned a position in the sequence.

The LB is first evaluated for the n classes of permutations, i.e., for those starting with 1, 2, ..., n, respectively. From the vertex with the lowest value, the LB is evaluated for the n-1 subclasses. From these values and the previously calculated values, the lowest vertex value is found, from which the LB is again evaluated. Proceeding in this manner and after a finite number of steps, a MAKESPAN LB* is found such that all the remaining vertices of the scheduling tree have values greater than LB*. The permutation with this label LB* gives an optimal solution to the scheduling problem.

Some preliminary results will now be discussed. A total of 69 problems have been solved by all three methods. The problems were all constructed with machine times ($t_{i,j}$) selected randomly from a uniform distribution of numbers from 1 to 50 for machine 1 and from 1 to 100 for machine 2.

First, a comparison of the average computational times for the three methods is shown in Figure 1. As expected, the branch-and-bound method required more time than either of the heuristics and is

not consistent with real-time control requirements, increasing rapidly as the number of jobs n increases. The highest average time for either heuristic was less than 0 1 second for n = 200, which is within real-time control constraints.

Next, heuristic Method 1 found the optimal solution in 66 percent of the cases, whereas heuristic Method 2 found the optimal solution in 45 percent of the cases. Heuristic Method 1 found solutions equal to or lower than Method 2 in 85 percent of the problems solved. However, in no case did the solutions by either heuristic method exceed the optimal solution by more than 3 percent. More work is needed to evaluate the effects of the distributions and ranges of machine times on the heuristic methods' efficiency and selection.

## CONCLUSION

A two-phase method for solving the resource allocation and scheduling problem of an adaptive control system such as that of a BMD system has been developed and is under evaluation. The allocation and scheduling have been placed into an analytical context from which it is possible to derive a dynamic methodology that can allocate BMD resources in an effective manner.

In Phase I, a nonlinear, constrained, search method was shown to solve a typical example of a BMD resource allocation problem involving search, interceptor tracking, and object tracking in less than 0.1 second. This solution time is well within the constraints for a real-time control BMD system.

In Phase II, the scheduling problem was structured as a two-machine problem with feedback so as to minimize total system processing time. Because of the often immense size of the problem, heuristic scheduling was employed, with two competing algorithms developed. A preliminary investigation of the efficiency of the algorithms was performed by using a branch-and-bound algorithm for finding the solution to restricted cases. The investigation indicates that the heuristic method solutions and computational times are consistent with real-time control requirements.

It is concluded that the scheme that has been developed, which gives an analytical approach for structuring and relating resource functions, is adequate for real-time performance criteria. Results to-date indicate the practicality of utilizing this scheme under the stringent real-time requirements of a typical BMD system.

## REFERENCES

1. G. Beveridge and R. Schechter, Optimization: Theory and Practice, New York: McGraw-Hill Book Company, 1970.

2. Z. A. Lomnicki, "A 'Branch-and-Bound' Algorithm for the Exact Solution of the Three-Machine Scheduling Problem," Opnal. Res. Quart. 16, pp. 89-100.

3. R. J. Giglio and H. M. Wagner, "Approximate Solutions to the Three-Machine Scheduling Problem," Opns. Res. 11, pp. 972-989.

4. R. D. Smith and R. A. Dudek, "A General Algorithm for Solution of the n-Job, M-Machine Sequencing Problem of the Flow Shop," Opns. Res. 15, pp. 71-82.

5. E. Ignall and L. Schrage, "Application of the Branch-and-Bound Technique to Some Flow-Shop Scheduling Problems," Opns. Res. 13, pp. 323-325.

6. J. N. D. Gupta, "A General Algorithm for the n X M Flowshop Scheduling Problem," The International Journal of Production Research, Vol. 7, No. 3.

7. H. H. Greenberg, "A Branch-Bound Solution to the General Scheduling Problem," Opns. Res. 16, pp. 353-361.

8. J. P. Ignizio and J. N. D. Gupta, Operations Research in Decision Making, New York: Crane, Russck and Company, Scheduled for Publication in 1974.

## TABLE 1. PROBLEM RESULTS

| NO. OF OBJECTS THREATENING, n | RADAR SEARCH RATE, F | TRACK RATE FOR OBJECTS, K | TRACK RATE FOR INTERCEPTORS, L | PROBABILITY OF RADAR SURVIVAL, Z | SOLUTION TIME (sec) |
|---|---|---|---|---|---|
| 1 | 6.5 | 13.5 | 23.6 | 0.937 | 0.032 |
| 2 | 4.7 | 10.3 | 18.5 | 0.846 | 0.043 |
| 3 | 3.7 | 8.6 | 12.3 | 0.706 | 0.036 |
| 4 | 3.0 | 7.3 | 9.3 | 0.521 | 0.028 |
| 5 | 2.4 | 6.4 | 7.4 | 0.336 | 0.073 |
| 6 | 2.1 | 5.7 | 6.2 | 0.191 | 0.035 |
| 7 | 1.8 | 5.1 | 5.3 | 0.096 | 0.035 |
| 8 | 1.5 | 4.6 | 4.6 | 0.043 | 0.053 |
| 9 | 1.3 | 4.2 | 4.1 | 0.018 | 0.028 |
| 10 | 1.2 | 3.8 | 3.7 | 0.007 | 0.024 |

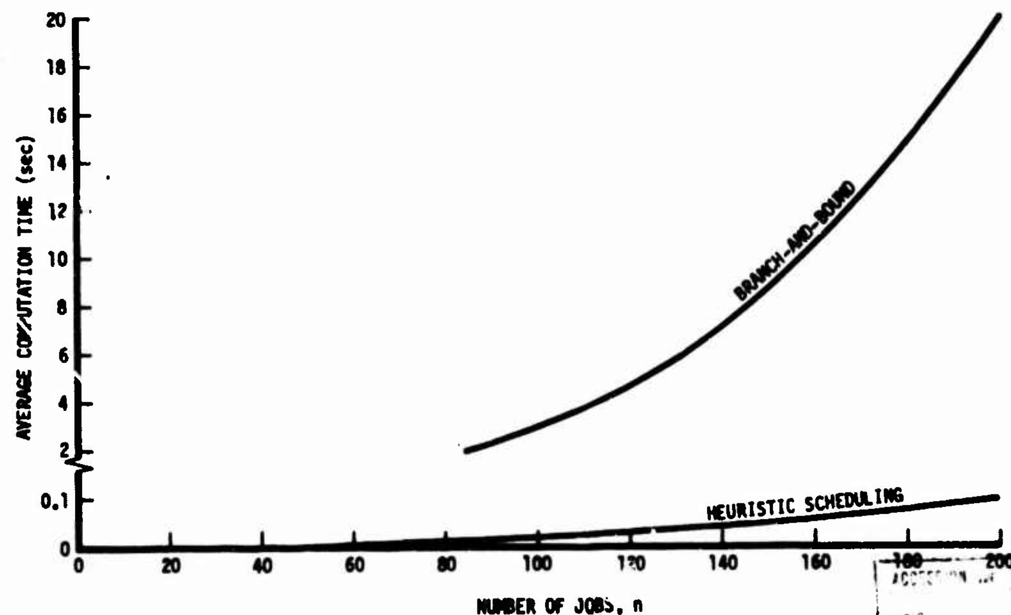


FIGURE 1. COMPUTATION TIME REQUIREMENT AS A FUNCTION OF THE NUMBER OF JOBS